Vysoká škola báňská – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Implementace technologie VOIP do systému inteligentního domu

VOIP Technology Implementation for Intelligent House

2014 **Marek Kovář**

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání bakalářské práce

Student: **Marek Kovář**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2601R013 Telekomunikační technika

Téma: Implementace technologie VOIP do systému inteligentního domu
VOIP Technology Implementation for Intelligent House

Zásady pro vypracování:

1. Definujte minimální požadavky na systém iDům pro implementaci VOIP.
2. Realizujte na systému iDum pobočkovou ústřednu pomocí programu Asterisk.
3. Nakonfigurujte Asterisk, aby pro komunikaci používal zvukovou kartu řídícího PC.
4. Vytvořte skripty pro vyzvánění a spojení pomocí prvku "Simplex".
5. Vytvořte skripty pro vyzvánění a spojení pomocí prvku "Telefon".
6. Vytvořte skripty pro spojení pomocí prvku "Tablo".
7. Vytvořte skripty pro spojení volání z prvku "Tablo" na předdefinované číslo.
8. Nakonfigurujte systém pro funkci ústředny pro spojení s ostatními prvky v síti používající VOIP.
9. Systém otestujte a zdokumentujte.

Seznam doporučené odborné literatury:

Manuálová příručka iDům.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Jiří Vychodil**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2014

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení studenta

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě dne: *7. května 2014*

.....
podpis studenta

Poděkování

Rád bych poděkoval Ing. Jiřímu Vychodilovi za odbornou pomoc a konzultaci při vytváření této bakalářské práce. Také bych rád poděkoval Ing. Michalu Jahelkovi Ph.D. za odbornou pomoc při vytváření skript pro iDům a obeznámením s funkcemi iDomu.

Abstrakt

Bakalářská práce se zabývá implementací technologie VOIP do systému inteligentní dům. Jeden druh inteligentního domu (iDům) je vyvíjen na katedře telekomunikační techniky Vysoké školy báňské – Technické univerzity Ostrava ve spolupráci s ELEKTRO-FA. PAVELEK, s.r.o. Cílem této bakalářské práce je určit minimální požadavky pro implementaci VOIP technologie do iDomu. Technologie VOIP bude v iDomu realizována pomocí programu Asterisk, který bude nakonfigurován tak, aby pro komunikaci používal zvukovou kartu řídícího PC. Program Asterisk bude komunikovat s prvky iDomu a to Simplex, Telefon a Tablo. Celá pobočková ústředna bude otestována a zdokumentována.

Klíčová slova

Inteligentní dům; iDům; komunikační technologie; Asterisk; technologie VOIP; pobočková ústředna.

Abstract

My bachelor thesis is focused on implement VoIP technology to the intelligent house called iDům (iHouse) which was developed at VŠB-Technical University of Ostrava on Department of Telecommunications in cooperation with the ELEKTRO-FA. PAVELEK, s.r.o. First objective of this bachelor thesis is describe minimal requirements for implement VoIP technology into iHouse. For implement this technology is used open source application Asterisk which will be using sound card of controller computer. The Asterisk will be able to communicated with elements of iHouse like Simplex, Tablo and Telefon (Telephone). The complete PBX will be tested and described.

Key words

Intelligent house; iDům; iHouse; communication technology; Asterisk; VoIP technology; PBX (Private Branch Exchange).

Seznam použitých zkratek

Zkratka	Význam
VoIP	Voice over Internet Protocol
PBX	Private Branch Exchange
PC	Personal Computer
TCP/IP	Transmission Control Protocol / Internet Protocol
PCM	Pulse code modulation
LCD	Liquid-crystal display
HDD	Hard Disk Drive
SIP	Session Initiation Protocol
IAX	Inter Asterisk eXchange
MGCP	Media Gateway Control Protocol
VoFR	Voice over Frame Relay
HTTP	HyperText Transfer Protocol
SMPT	Simple Mail Transfer Protocol
NAT	Network Address Translation
CLI	Command Line Interface
QOS	Quality of Service
RTP	Real-Time Transport Protocol
IVR	Interactive Voice Response
ITU-T	International Telecommunication Union – Telecommunication Standardization Sector
WAV	Waveform audio file format

Obsah

Úvod

Tato bakalářská práce pojednává o technologii VoIP a její implementaci do systému iDům. Inteligentní dům je takový dům, kde jsou aplikovány různé nastavení spotřebičů a zařízení, které systém inteligentního domu sdružuje pro komfortnější a pohodlnější žití.

Na VŠB-TU Ostrava společně s firmou ELEKTRO-FA PAVELEK s.r.o. je druh takového inteligentního domu vyvíjen, a to na katedře telekomunikační techniky. Tento konkrétní inteligentní dům si dává za cíl použití levných koncových zařízení a byl pro něj použit název iDům.

Cílem mé bakalářské práce je implementovat technologii VoIP do inteligentního domu, což znamená vytvoření pobočkové ústředny pomocí programu Asterisk. Nastavení a přizpůsobení tohoto programu k tomu, aby mohl komunikovat s ostatními zařízeními používající tuto technologii a komunikoval se zvukovou kartou řídícího PC. Dále zpřístupnit spojení mezi jednotlivými prvky v inteligentním domě a to Simplex, Telefon a Tablo. A jako poslední nadefinovat volání z prvku Tablo na předdefinované číslo.

Postup mé bakalářské práce bude následovný. V první řadě seznámím čtenáře s teorií ohledně systému iDům a technologií VoIP. V druhé řadě určím z teoretických poznatků, jaké jsou minimální požadavky pro implementaci technologie VoIP do inteligentního domu. Poté čtenáře seznámím s programem Asterisk a jeho fungováním, nastavováním a vytvoření pobočkové ústředny v systému iDům. Následovat bude zprovoznění programu iDům na PC, připojeném k rozhraní iDomu, kde budu testovat funkčnost již vyvinutých prvků. Dále odzkouším jednotlivé vytvořené skripty. Z poznatků již vytvořených skript vytvořím skripty pro spolupráci s jednotlivými prvky iDomu jako jsou Simplex, Tablo a Telefon. Následovat bude nastavení programu Asterisk pro spolupráci se zvukovou kartou řídícího PC. Posledním krokem bude odzkoušení funkčnosti na systému iDům. Doladění jednotlivých nedostatků a závěrečné testování a zhodnocení.

V úvodu bych vás rád seznámil s obsahem jednotlivých kapitol. Než dojdeme k samotné implementaci této technologie, seznámíme se nejprve s technologií VoIP a dále se systémem iDům. Proto první kapitola bude věnována seznámením s technologií VoIP a systémem iDům.

V následující kapitole si objasníme minimální požadavky na systém iDům pro implementaci technologie VoIP a také minimální požadavky pro program Asterisk.

Ve třetí kapitole si popíšeme konfiguraci programu Asterisk pro spolupráci se zvukovou kartou řídicího PC.

Čtvrtá kapitola bude věnována vytvořeným skriptům pro vyzvánění a spojení pomocí prvku „Simplex“, pro vyzvánění a spojení pomocí prvku „Telefon“, pro spojení pomocí prvku „Tablo“ a pro volání z prvku „Tablo“ na předdefinované číslo.

Pátá kapitola nás obeznámí s konfigurací systému pro funkci ústředny pro spojení s ostatními prvky v síti používající VoIP.

Závěrečná kapitola vše shrne a zhodnotí otestovaný systém.

1 Seznámení s technologii VoIP a systémem iDům

Dříve než přistoupíme k podrobnějšímu řešení problému implementace, seznámíme se nejprve s technologií VoIP a následně se samotným systémem iDům.

Technologie VoIP dnes již může hrdě nahradit klasickou telefonní přípojku díky svým mnohým výhodám, od kvality hovoru až po velice přijatelnou cenu i na delší vzdálenosti. Dnes v moderním světě není problém k zapojení této technologie do domácností, protože již skoro každý má doma internet. Přes VoIP jde volat pomocí klasického pevného telefonu a nyní i pomocí chytrých mobilních telefonů nebo z PC za pomocí různých programu (např. Skype), sluchátek a mikrofonu.

Systém iDům nám dává možnost modernějšího, komfortnějšího a hlavně úspornějšího bydlení než dosavadní klasické elektroinstalace. Dává nám možnost si zpohodlnit život a neobtěžovat se s častým nastavováním jednotlivých spotřebičů, kdy můžeme jednotlivé nastavení zautomatizovat.

1.1 Technologie VoIP

VoIP je zkratkou pro Voice over Internet Protokol neboli přenos hlasu přes IP protokol. VoIP komunikuje přes datové sítě s protokolem TCP/IP. Laicky by se dala tato technologie definovat jako „volání přes internet“. Počátek VoIP sahá do roku 1995, kdy byl umožněn první hovor přes IP protokol pomocí programu vyvíjeného firmou Vocaltec. Od té doby se o VoIP začaly zajímat spousta jiných firem, protože VoIP se stal velice lukrativní technologií pro hovory. Stal se levnější metodou pro meziměstské a mezinárodní hovory a dopracoval se do alternativy pro klasické telefonní přípojky.

Velkou výhodou IP protokolu je, že je implementován skoro všude a to díky rozšíření a oblibě Internetu. A z toho plyne i výhoda technologie VoIP, která se teoreticky dá provozovat všude, kde je IP protokol a to i bez ohledu na to, jakou máme přípojku (ADSL, kabel…). Samozřejmé záleží na kvalitě a parametrech přípojky.

Naopak hlavní nevýhodou VoIP je zpoždění. Další nevýhodou je pak ztráta dat při přenosu, ale většinou to jsou data tak malá, že to lidské ucho ani nepostřehne, tedy pokud není výpadek dat v jednom místě větší. Co se týče problému zpoždění, ten by se dal nazvat hlavní nevýhodou, což může způsobovat to, že při hovoru se zdá, že druhá strana nereaguje a daná věta volajícího je zopakována a to způsobí, že si dva volající skáčou do řeči.

Princip cesty hlasu v technologii VoIP (vidíme na obrázku 1.1) spočívá v převodu hlasu na binární signál, což je provedeno digitalizací hlasu (řeči), dále dojde ke kompresi (zmenšení objemu dat) a rozdělení na pakety, které jsou posílány

prostřednictvím sítě s ostatními daty, které posílá PC. Na přijímací straně pak dochází k přesně opačnému procesu, kdy jsou pakety sloučeny, dekomprimovány a dedigitalizovány, díky čemuž získáme zpět původní data. Pro technologii VoIP je minimální přenosové pásmo o velikosti 16 kbps (při velké kompresi i 6 kbps), naproti tomu klasická digitální linka musí mít k přenosu dat rychlost 64 kbps. K převodu hlasu (spojitého signálu) do digitální podoby (nespojitého signálu) se nejčastěji používá PCM modulace, která využívá Shannonova-Kotelnikovova zákona.

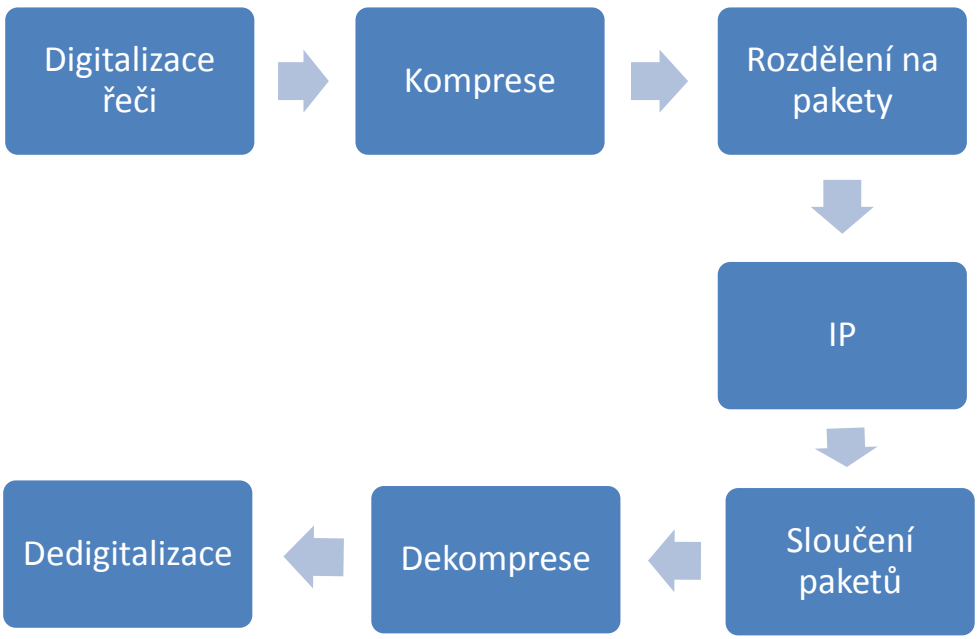
Faktorů ovlivňující kvalitu hovoru je několik, například:

- Kodek
- Technické řešení
- Způsob využití datové přípojky

Kodek je vlastně algoritmus pro digitalizaci a kompresi, u nás konkrétně hlasu, respektive převod analogového hlasu na digitální data. Hlavními parametry kodeků je velikost digitalizovaných dat a kvalita digitalizovaného signálu. V poměru největší kvalita signálu ku co nejmenší velikost dat.

Konkrétní technické řešení. Technická řešení se můžou lišit u proprietárních řešeních a řešeních postavených na standardech (H.323, SIP nebo IAX).

Způsob využití datové přípojky. Zde jsou ovlivňujícími faktory, zda je přípojka využívána souběžně i s jinými uživateli, nebo zda se přípojka používá i pro jiné souběžné úkony, jako například stahování z internetu apod.[3].



Obrázek 1.1: Cesta hlasu v technologii VoIP

1.2 VoIP Protokoly

Ustálené mechanismy či standardy pro zprostředkování připojení mezi dvěma koncovými body se nazývají protokoly. Laicky řečeno, protokol je nějaké pravidlo, podle kterého se určuje, jak se data budou přenášet a jak se mají chovat, aby si obě koncová zařízení rozuměla a dokázala se domluvit. SIP, H.323 a IAX jsou hojně používané protokoly ve VoIP a my si je nyní trochu přiblížíme.

1.2.1 SIP

SIP je zkratkou pro Session Initiation Protocol. Jedná se o průlomový protokol. Je poměrně jednoduchý se syntaxí podobnou rodinným protokolům jako je HTTP a SMTP. Je založen na end-to-end koncepci (logika je na koncových zařízeních) a nestará se o přenos hlasu. Pro přenos hlasu se používá jiný protokol, nazvaný RTP, což je zkratka pro Real-Time Transport Protocol. V Asterisku je SIP podporován chan_sip.so modulem.

SIP využívá pro komunikaci dobře známý port 5060, nejčastěji využívá UDP, ale lze také využít TCP. SIP se obhajuje jako dobrý protokol pro VoIP. Všichni noví uživatelé a nové produkty vyžadují podporu SIP. SIP je jeden z hlavních protokolů pro hlasové aplikace pro následujících pár let (možná i déle).

Největší problémem protokolu SIP je problém s NAT. Jelikož SIP nepoužívá stejný port pro přenos zvuku jako pro signalizační část, může docházet k navázání spojení mezi dvěma SIP zařízeními, avšak audio výstup není srozumitelný, nebo není srozumitelný z jedné strany hovoru. Tento problém však lze řešit pomocí směrování portů na směrovači (router). Díky této skutečnosti poskytovatelé VoIP služby dodávají celý router, již s potřebným nastavením, aby tomuto problému předešli.

1.2.2 H.323

H.323 je podobným protokolem jako protokol SIP. H.323 poskytuje větší podporu pro videokonference a také proto se stal jejím standardem. Prozatím je to nejpoužívanější VoIP protokol. Problémem stejně jako u SIP je implementace s NAT. Jelikož tento protokol je binární, je odstranění potíží s NAT složitější. Velký přínos tomuto protokolu dodává QOS, což je technologie přispívající k lepším zvládání definice a dodržení síťové politiky, také zlepšuje hlasové a multimediální přenosy, což je taky důvod, proč se H.323 stal standardem pro videokonference.

1.2.3 IAX

IAX reprezentuje zkratku pro Inter-Asterisk eXchange protokol. Je nejmladším protokolem používaný pro VoIP. Nyní už existuje druhá verze, tedy IAX2, avšak

oficiálně, ať už mluvíme o „IAX“ nebo „IAX2“, jedná se o stejnou (nejnovější) verzi. Tento protokol není zatím tak rozšířený jako jiné VoIP protokoly. Jedná se o otevřený protokol, který si může kdokoliv stáhnout a vyvíjet. IAX využívá UDP a je průchodný přes NAT, na rozdíl od H.323 a SIP, což v podstatě odstraňuje problém s nastavením NAT. IAX je díky svým vlastnostem rychle šířící se protokol pro VoIP poskytovatele a pro výrobce VoIP zařízení [6].

1.3 Systém iDům

iDům je systém inteligentního domu. A jak už kořen slova „inteligentní“ napovídá, jedná se o systém, který je inteligentní. Je schopen sám pracovat a plnit jednotlivé úkony podle naprogramování či nastavení obsluhy. Pojem inteligentní dům chápejme jako klasický dům, který je ovládán elektronikou, díky které je žití v tomto domě pohodlnější a samotný provoz domu je energeticky úspornější. Jedná se tedy o uživatelsky výhodnější a pohodlnější možnost, než klasická elektroinstalace. Nabízí uživateli dokonalé řízení jednotlivých prvků. Například osvětlení, topení nebo klimatizace v bytě, telefonu aj..

Inteligentní dům se dá také chápat jako systém propojení většiny elektrospotřebičů do jednoho společného mechanismu, který je ovládán různými ovládacími prvky. Ovládacími prvky inteligentního domu mohou být jak mobilní telefony, tak tablety, či přístup přes internetovou aplikaci aj.. Jelikož v současné době je spíše poptávka po pohodlnějším řešení, je ovládání mobilním telefonem nejvíce žádané, což není divu, protože většina z nás má vždy mobilní telefon u sebe, takže je to velice lukrativní řešení. Ovládání inteligentního domu pracuje na společném mechanismu, který propojuje jednotlivé zařízení a spotřebiče. V naprogramovaných, či nastavených krocích jsou jednotlivé zařízení a spotřebiče spouštěny, nastavovány nebo vypínány.

Nejčastějším nastavením uživatele bývá klimatizace a topení. Jako příklad si můžeme uvést nastavení topení, kdy se v určitou hodinu nastaví teplota vyhřívání v místnosti na méně stupňů a při příchodu je teplota, chvíli před vaším příchodem, nastavena zpět na více stupňů a tudíž přijdete do již vyhřátého domu. Princip tohoto příkladu nejen šetří energii, ale i vám zpohodlní život, kdy nemusíte topení zapínat až po příchodu. Dalšími nastaveními může být ztlumení topení při otevření okna a mnoho dalších. Topení a klimatizace není jediným nastavením, kterého v inteligentním domě můžeme dosáhnout. Můžou se zautomatizovat ranní budíčky, kdy vás inteligentní dům vzbudí, otevře rolety, zapne podlahové topení, zapne ohřev vody na čaj, zapne kávovar a mnoho dalších.

Jak vidíte z příkladů, které jsem uvedl, inteligentní dům dává širokou škálu možností, jak si ho nastavit a přizpůsobit. Může se i přizpůsobit vaším pravidelným zájmům či činnostem, například zapnutí vyhřívání sauny nebo vířivky.

Náš konkrétní iDům je systém inteligentního domu, který je vyvíjen na katedře telekomunikační techniky Vysoké školy Báňské – Technické Univerzity v Ostravě ve spolupráci s firmou ELEKTRO-FA. PAVELEK, s.r.o.. Tento iDům má za cíl vytvořit inteligentní dům s levnými koncovými zařízeními a také s nízkým odběrem proudu.

Systém také může pracovat ve dvou módech:

- systém bez řízení,
- systém s centrálním řízením

1.3.1 Systém bez řízení

Jak už z názvu vyplývá, bude se jednat o jednodušší a tudíž i levnější variantu systému. Tento druh systému se může použít pro jednodušší systémy, které nepotřebují složité ovládání, jako jsou jednoduchá zabezpečení či řízení osvětlení. Tyto jednotlivé systémy lze zkombinovat dohromady či rozšířit o jiná řízení, což však musí již udělat kvalifikovaná osoba, nikoliv uživatel. Kvalifikovaná osoba může zasáhnout do konfigurace pomocí iDům konfigurátoru a PC s potřebným hardwarem, což nám dává velkou nevýhodu při přeprogramování systému.

U systému bez řízení má každé zařízení v sobě uloženou adresu, či seznam adres, na které posílá své informace. Adresa také může být volena uživatelem (na ovládacím prvku – např. volba klávesy na domovním tablu).

1.3.2 Systém s centrálním řízením

Jádrem systému s centrálním zařízením je miniaturní nízkospotřebové PC běžící na operačním systému Linux. Na rozdíl od systému bez řízení má uživatel možnost nastavit jednotlivé parametry, či změnit chod celého systému. PC má možnost připojení jednotlivých periférií od LCD display až po touchpady. Jak již bylo řečeno, lze tento systém ovládat různými ovládacími prvky (mobilní telefon, touchpad, klávesnice a monitor apod.). Operační systém je spuštěn z paměti FLASH, která je pouze pro čtení, což znamená, že na ní nelze zapisovat data. Ve vnější zálohované paměti RAM jsou ukládány informace a data o zapnutých zařízeních a další parametry, takže v případě krátkodobé ztráty napájení nejsou data ztracena. V případě potřeby lze počítač obohatit o HDD, kde lze ukládat například záznamy z kamer, historii volání apod.

Všechny prvky inteligentního domu jsou propojeny čtyřvodičovou sběrnicí (napájecí vodiče plus a mínus, datový vodič D – pro přenos signalizace a audio vodič A – pro přenos specifické informace jako video či video), což může znamenat, že

nefunkčnost jednoho prvku způsobí nefunkčnost celého systému (ostatní prvky na sběrnici nejsou zničeny). Při odstranění nebo výměně nefunkčního zařízení (spotřebiče) bude systém správně fungovat dál dle předchozích nastavení. Chyby na zařízeních může způsobit i pokles napětí na sběrnici, které může nastat například při uvolněném kontaktu.

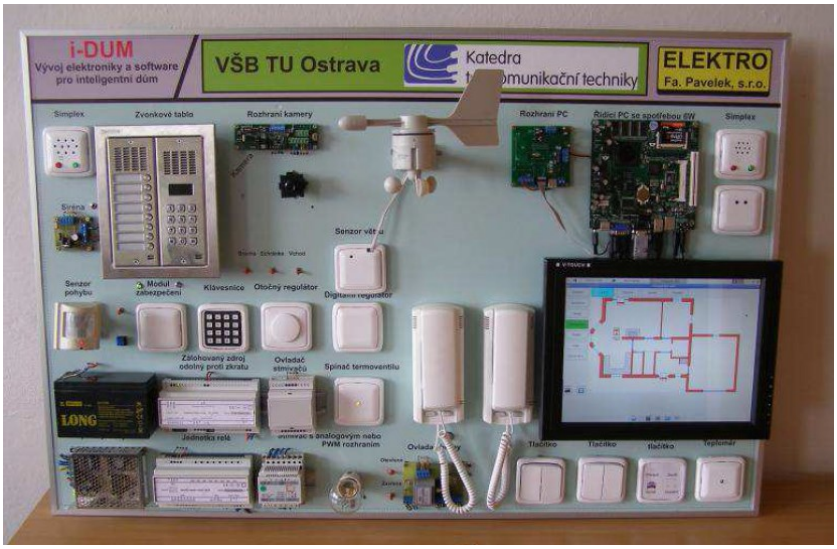
Pro komunikaci se používá bezestavový protokol a data jsou vysílány po 8 bitech.

Před přenosem mají všechny prvky status spánek pro zajištění nízké spotřeby. Proto, pokud chceme odesílat data musíme provést sekvenci probuzení z režimu spánku. Tento budící signál má 5 ms v logické 0 a 1 ms v logické 1. Poté následuje protikolizní ochrana zajišťující vysílání jen jednoho zařízení. Dále se vysílá zpráva.

Zprávy mají různou délku. Hlavička je vždy vysílána jako první a je stejná. Obsahuje informace:

- o typu zprávy,
- o cílovém zařízení,
- o zdrojovém zařízení,
- délka zprávy, což je počet bytů celé zprávy včetně hlavičky,
- uživatelské informace,
- tělo – proměnné velikosti,
- kontrolní součet (CRC).

Mikropočítače jsou pomalé, proto pokud je přerušení vysílání od přijatého posledního bitu většího než 1,5 ms (konkrétně jsou překročeny 2 ms), je spojení považováno za ukončené a k sběrnici mohou být připojena jiná zařízení [1].



Obrázek 1.2: Demonstrační panel iDům

2 Minimální požadavky na systém iDům pro implementaci VoIP

Jelikož VoIP, jak už bylo zmíněno, je technologie využívající IP protokol pro přenos hovoru, je nutné zajistit trvalý přístup na internet (nebo jiné datové sítě podporující protokol TCP/UDP). Minimální rychlost připojení by měla být 128 kb/s, pokud by byla nižší, může docházet k výpadku spojení, zpoždění atd.. Avšak záleží na mnoha jiných aspektech, jako jsou kodeky, přípojka apod., takže i tato minimální rychlost může být spekulativní.

Systém iDům pracuje na operačním systému Linux s distribucí Debian, což můžeme pokládat za ideální volbu operačního systému. Linux je stabilní a proto vhodný pro zázemí jakéhokoliv systému. Proto také více jak 90% superpočítačů má základ právě na Linuxovém prostředí. Dokonce jedna z nejrozšířenějších mobilních platforem je založena na Linux, což je nám známý Android.

Na iDomu bude vytvořena pobočková ústředna, která bude vyžadovat instalaci programu Asterisk. Co se týče minimálních požadavků pro Asterisk, tak toto téma je velice diskutabilní. Záleží totiž na množství požadavků, kterými budeme dále naši ústřednu vyvíjet. Pokud by zůstalo u mých nastavení, a to pro hovor na iDům, možnost hovoru z iDomu, případně VoIP hovor mezi klienty, či hovor na iDům z jiného Asterisku, tak požadavky na Asterisk by nebyly velké. Nejmenší možné požadavky na PC, kdy byl Asterisk rozběhnut byly Pentium 100Mhz, 32MB RAM. Jak vidíme, požadavky jsou opravdu minimální. Řídicí systém navíc řídí program iDům, proto by toto řešení nebylo ideální. Obraťme se k reálnějšímu řešení a požadavkům, aby byl Asterisk schopen pracovat i s řídícím programem iDomu. Minimálními požadavky by mohly být 512MB RAM se systémem, například Ahtlon 1,5 až 2,4 GHz [12].

Dále máme požadavky na vhodné periférie, které chceme do pobočkové ústředny zapojit. Jelikož iDům už určité periférie má, bude se jednat konkrétně o Simplex, Tablo a Telefon. Dále použijeme softwarové telefony pro ověření funkčnosti volání na Asterisk, na iDům a následné ověření funkčnosti trunku mezi dvěma Asterisky.

3 Asterisk

Dříve, než přistoupíme ke konkrétní konfiguraci programu Asterisk, nastíníme si nejprve co to program Asterisk je.

Asterisk je počítačový software sloužící jako ústředna (PBX) přímo na vašem PC. Jedná se o open source aplikaci (otevřený zdrojový kód) běžící na Unix a Linux. Může běžet i na jiných operačních systémech, ale hlavní vývojářskou platformou je Linux. Poskytuje nám všechny vlastnosti klasické pobočkové ústředny.

My se soustřeďme na podporu VoIP v Asterisku. Cílem systému je, aby podporoval současné i budoucí telefonní technologie. Obecně se rozhraní rozdělují do 3 základních skupin:

- Zaptel hardware
- Non-Zaptel hardware
- Packet voice

Nás bude zajímat Packet voice. K přenosu hlasu dochází v komunikaci přes paketovou síť (IP a Frame Relay) a také jsou to jediná rozhraní, která nepotřebují specializovaný hardware. Autor tohoto programu, Mark Spencer, neměl rád patentovaný protokol H.323, proto se rozhodl vytvořit vlastní. Vznikl tak protokol IAX. Zprvu byla komunikace mezi dvěma body pouze v podpoře protokolu IAX na obou koncích. Posléze však byla implementována podpora pro součinnost s jinými VoIP systémy a podpora dalších packet voice protokolů, jako například zmiňovaný H.323, nebo SIP, MGCP a VoFR.

Asterisk vystupuje svou architekturou jako jednoduchý systém. Jedná se v podstatě o středový prvek spojující jednotlivé strany (telefonní technologie jedné strany s telefonními aplikacemi strany druhé). Telefonní technologie mohou být buď VoIP služby (SIP, H.323, IAX, Cisco skinny...), nebo tradiční TDM technologie (ISDN PRI, ISDN BRI, POTS…). Telefonními aplikacemi mohou být přesměrování hovoru, hlasové schránky, hlasové pošty, automatické obsluhy, interkom apod..

Jelikož se Asterisk snaží být přizpůsobitelný okolo jádra PBX, obsahuje nadefinované specifické API, které ovládá vnitřní přípojka PBX (konkrétně určité protokoly, kodeky a hardwarové rozhraní telefonních aplikací).

3.1 Dialplan

Dialplan neboli číslovací plán se dá pokládat za srdce Asterisk systému, definuje ovládání a směrování příchozích a odchozích hovorů. K tomu, abychom správně a efektivně pracovali se systémem Asterisk je bezprostředně nutné pochopit dialplan.

Dialplan je specifikovaný v konfiguračním souboru extensions.conf. Cesta k tomuto souboru bývá obvykle /etc/asterisk/directory, ale samozřejmě záleží na instalaci Asterisku, takže cesty k souboru se mohou lišit.

Organizačně se diaplan rozděluje na:

- contexts
- extensions
- priorities
- applications

3.1.1 **Context**

Dialplan je složen ze sekcí nazývaných context. Context je skupina několika extensions. Context se píšou do hranatých závorek ([ ]). Například context pro příchozí hovor je definován jako [incoming]. Context udržuje jednotlivé části dialplanu. Extension definovaný v jednom context je zcela izolovaný od extension v jiném context. Na začátku dialplanu jsou definovány dva speciální context nazvané [general] a [globals].

[Generals] obsahuje hlavní nastavení dialplanu.

[Globals] obsahuje globální proměnné.

Context můžeme libovolně pojmenovat, kromě názvu globals a generals. Definované context jsou nejdůležitější části konfiguračního souboru extensions.conf a také celé konfigurace systému.

3.1.2 **Extensions**

Extensions je definice jednotlivých unikátních kroků (každý krok obsahuje aplikaci), které Asterisk bude volat. Syntaxí je:

```
exten =>
```

Kompletní extension je složen z 3 základních částí:

- jméno (název) nebo číslo extension,
- priority,
- aplikace nebo příkazu, které definují nějaké akce, které budou volány.

Tyto tři základní části jsou složeny následovně:

```
exten => jméno,priorita,aplikace()
```

Konkrétní příklad může být například extension pojmenovaná „123“ s prioritou 1 a aplikací Answer () :

```
exten => 123,1,Answer()
```

3.1.3 **Priority (Priorities)**

Každá extension může obsahovat více kroků, proto se píšou priority, které určí, co se bude vykonávat jako první. Tudíž extension s prioritním číslem 1 bude vykonán jako první. Jako příklad se může uvést extension Answer a Hangup, kdy první je hovor vyzvednut (Answer s prioritou 1) a poté položen (Hangup s prioritou 2):

```
exten => 123,1,Answer()
exten => 123,2,Hangup()
```

Pokud bychom však pokaždé ručně zadávali čísla priorit, mohla by nastat následující problém. Představte si, že máte extension s 15 prioritami a potřebujete napsat další extension na druhou pozici, tím pádem byste museli přepisovat všechny priority o jedno číslo nahoru a vzniklo by tak 16 extension. Pro tento případ existuje tzv. neočíslovaná priorita, což je značeno písmenem n. Takto označena priorita říká Asterisku, přiřaď za písmeno n číslo o velikosti předchozí velikosti čísla plus jedna, což znamená, že pokud předchozí číslo priority bylo 5 a další extension bude mít prioritu označenou n**,** velikost této priority bude 6. Konkrétní příklad:

```
exten => 123,1,Answer()
exten => 123,n,Udělej něco
exten => 123,n,Udělej něco jiného
exten => 123,n,Hangup()
```

Na příkladu vidíme, že u první priority je napsané číslo 1 a u dalších už pouze n. Asterisk si už sám přiřadí čísla priorit. Důležitá věc při neočíslovaných prioritách je ta, že u první extension musíme napsat číslo priority 1, pokud tak neučiníme, tak extension nebude dostupná.

3.1.4 **Aplikace (Applications)**

Aplikace jsou „pracovníkem“ celého dialplanu. Určují a specifikují jednotlivé akce na konkrétním kanále. Může to být přehrání zvuku, přijmutí stisknutého tlačítka, zavěšení hovoru atd.. Některé aplikace nepotřebují ke své činnosti více instrukcí jako například Hangup() nebo Answer(). Avšak některé potřebují více instrukcí, tyto instrukce pro jednotlivé aplikace nazýváme argumenty [5][6][11]..

3.2 Makra

Další funkcí Asterisku jsou makra. Pomáhají přehlednosti kódu, ale nejen to. Hlavní funkcí maker je zjednodušit psaní dialplanu. Pokud některé kódy máme vícekrát je použití maker na místě. Respektive, pokud například máme 100 uživatelů na naší pobočkové ústředně, tak by trvalo mnoho času, než bychom nastavili jednotlivě každého uživatele, proto použijeme makra. Strukturu si připravíme do makra a pomoci jednořádkového příkazu zavoláme makro a přiřadíme mu proměnné konkrétního uživatele. Makra jsou volány jedním řádkem, ale funkce makra mohou být obsažena na více řádcích. Pro bližší představu konkrétní příklad:

```
[macro-prehraj]
exten => s,1,Answer()
exten => s,n,Playback(${ARG1})
exten => s,n,Wait(3)
exten => s,n,Hangup

[default]
exten => _101,1,Macro(prehraj,/var/lib/asterisk/sounds/1)
exten => _102,1,Macro(prehraj,/var/lib/asterisk/sounds/2)
exten => _103,1,Macro(prehraj,/var/lib/asterisk/sounds/3)
exten => _104,1,Macro(prehraj,/var/lib/asterisk/sounds/4)
```

Na příkladu vidíme, že nemusíme vypisovat všechny 4 hovory zvlášť, ale stačí základní syntaxe, kde jednotlivě vkládáme proměnné pro každé číslo. Zde konkrétně: hovor se přijme, přehraje se zvuk, podle volaného čísla, Asterisk 3 sekundy počká a hovor zavěsí.

Jak jste již z příkladu zjistili, funkce maker s širokým využitím je velice užitečná. Zvláště pak ušetří čas, při přidávání nové extension do dialplanu.[11]

3.3 Instalace Asterisku

Asterisk nainstalujeme jednoduchým příkazem v terminálu a to:

- ***sudo apt-get install asterisk***

Po instalaci nás Asterisk vyzve k nastavení ITU-T telefonního kódu. Pro Českou republiku je to číslo 420, které známe při zadávání čísla, pokud chceme volat ze zahraničí do České republiky.

Pokud po instalaci chcete ověřit, že Asterisk běží, stačí zadat:

- ***sudo asterisk –r,***

čímž se dostaneme do CLI Asterisku a můžeme začít konfigurovat.

Konfigurační soubory k Asterisku najdeme ve složce /etc/asterisk.

Je také možné, že při instalaci vám terminál vypíše, že Asterisk nemá žádné instalační balíčky, což bývá nejčastější chybou. Pokud k tomu dojde, nejspíš nemáte zpřístupněné některé instalační odkazy. Proto si otevřete v libovolném textovém editoru soubor source.list, naleznete jej v /etc/apt.

V tomto souboru použijte znak „#“ pro komentář. Tento znak vložte před:

```
deb cdrom:[Debian GNU/Linux 6.0.9 _ Squeeze_ - Official
i386 Binary-1 20140215-12:23] / squeeze main
```

Po vložení to bude vypadat:

```
#deb cdrom:[Debian GNU/Linux 6.0.9 _ Squeeze_ - Official
i386 Binary-1 20140215-12:23] / squeeze main
```

Pokud používáte jinou verzi Linuxu, samozřejmě se tento řádek bude lišit. Dále tam vložte následující řádky, většinou už tam jsou, jen bývají zakomentované. Proto, pokud je tam máte, pouze odstraňte znak komentáře (#). Řádky k přidání jsou následující:

```
deb http://http.debian.org/debian/ squeeze main contrib
non-free
deb-src http://security.debian.org/ squeeze/updates main
contrib free
```

Poté proveďte update systému, pomocí:

- ***apt-get install updates***

Po těchto úpravách by měl být balíček Asterisku dostupný a bez problému by měl být nainstalován. [6]

4 Realizace na systému iDům pobočkové ústředny pomocí programu Asterisk.

Asterisk již máme nainstalovaný (3.3 Instalace Asterisku). Nyní přistoupíme k samotné konfiguraci Asterisku, abychom ho nastavili jako pobočkovou ústřednu.

Prvním důležitým konfiguračním souborem pro nás bude sip.conf (etc/asterisk/sip.conf), kde nastavíme v [generals] port, který budeme používat, což bude 5060 (UDP nebo TCP používaný protokolem SIP). Bind address je IP adresa naší vnitřní síťové karty a je to IP adresa pro navázání spojení. My zadáme adresu 0.0.0.0, což nám říká, že se může připojit kdokoliv z jakékoliv IP adresy (pokud zná naši IP adresu). Konfigurace v sip.conf:

```
Bindadr=0.0.0.0
port=5060
context=idum
disallow=all
allow=alaw
allow=gsm
allow=ulaw
```

Context je název ústředny. Při prvním zapnutí bývá nastaven na jméno default. My jej změníme na idum, což konkrétně znamená, že pokud zavoláme na náš Asterisk, tak první context, který se otevře nebude [default], nýbrž [idum].

Allow je povolení jednotlivých kodeků, které budou využity při spojeních mezi telefony. Pokud chceme preferovat pouze některé kodeky, zakážeme pomocí *disallow=all* všechny kodeky a dále si nadefinujeme pouze preferované kodeky. Což bude vypadat následovně *allow=nazev_kodeku* (můžeme také nastavit u jednotlivých klientů).

Dále můžeme nakonfigurovat jednotlivé klienty. V našem případě to bude pouze jediný klient a to iDům, kterého použijeme pro pozdější testování. Tento klient bude klientem pro softwarovy telefon Ekiga, který použiji pro testování volání na Asterisk. V případě potřeby k nastavení dalších klientů vás odkážu do sip.conf, kde se jednotliví klienti nastavují.

```
[idum]
type=friend
mailbox=100@idum
auth=md5
username=idum
```

```
secret=idum
callerid=("idum" <100>)
host=dynamic
language=cz
```

Zde vidíte kód pro nastavení klienta s číslem pobočky 100. Context máme pojmenovaný idum.

Type nám říká vztah uživatele k Asterisku. Dalšími možnostmi mohou být: user, peer a friend.

Mailbox je určení hlasové schránky v případě nepřijetí hovoru klientem, kterou bychom museli případně nastavit.

Auth je nastavení šifrování.

Username je uživatelské jméno používané klientem při ověřování.

Secret je heslo uživatele, které uživatel používá při autentifikaci SIP služby.

CallerID je řetězec, který se bude zobrazovat na displeji volajícího a taky bude tímto řetězcem identifikovaný.

Host je nastavení buď dynamic, tzn., že se SIP telefon může přihlašovat z jakékoliv IP adresy k Asterisku. Dále se může zadat jméno či statická IP adresa [7].

Následujícím nastavením bude nastavení konfiguračního souboru extensions.conf, který se nachází v /etc/asterisk/. Zde si nakonfigurujeme dialplan pro IVR, který nás bude odkazovat na jednotlivé periferie iDomu, konkrétně na Simplex, Telefon a Tablo. Také nastavíme static a writeprotect na yes, čímž povolíme přepisování dialplanu z CLI Asterisku. Nastavení extensions.conf:

```
[general]
static=yes
writeprotect=yes


[idum]
exten => 100,1,Answer()
exten => 100,n,Background(/var/lib/asterisk/sounds/cz/main-menu)
exten => 100,n,WaitExten(15)

exten => 1,1,Playback(/var/lib/asterisk/sounds/cz/simplexP)
exten => 1,n,System(/iDum/SpustSkript VyzvaneniSimplexP.scr 0)
exten => 1,n,Dial(console/alsa)
```

```
exten => 2,1,Playback(/var/lib/asterisk/sounds/cz/simplexL)
exten => 2,n,System(/iDum/SpustSkript VyzvaneniSimplexL.scr 0)
exten => 2,n,Dial(console/alsa)

exten => 3,1,Playback(/var/lib/asterisk/sounds/cz/tablo)
exten => 3,n,System(/iDum/SpustSkript VolaniNaTablo.scr 0)
exten => 3,n,Dial(console/alsa)

exten => 4,1,Playback(/var/lib/asterisk/sounds/cz/telefon)
exten => 4,n,System(/iDum/SpustSkript VolaniNaTelefon.scr 0)
exten => 4,n,Dial(console/alsa)

exten => 5,1,Goto(idum,200,1)

exten => i,1,Playback(/var/lib/asterisk/sounds/cz/neplatna)
exten => i,n,Goto(incoming,100,1)

exten => t,1,Playback(/var/lib/asterisk/sounds/cz/naslysenou)
exten => t,n,Hangup
```

Nastavením extension jsme nastavili jednoduché hlasové menu (IVR), které uživatele přesměruje tam, kam sám uživatel chce. Po vyzvednutí hovoru nám přehraje úvodní slovo, kde nám řekne, jaká číslice co skrývá. Pokud stiskneme číslo jedna, Asterisk nás přesměruje na SimplexP pomocí zavolání externího skriptu, který má název VyzvaneniSimplexP.scr. V případě volby dvě nás Asterisk přesměruje na druhý Simplex, a to SimplexL. U čísla tři se nachází přesměrování na Tablo, u čtyřky přesměrování na Telefon a pětka nám obnoví spojení, což znamená, že znovu zavolá na toto číslo, čímž se hlasové menu znovu zopakuje. Pokud jsou číslice neplatně zadané, přehraje chybovou hlášku o neplatném čísle, k tomu slouží extension „i“. Po překročení časového limitu přehraje „na shledanou“ a hovor zavěsí.

Dále můžeme vytvořit echotest ústředny. Do dialplanu, do context idum nastavíme tyto extensions:

```
exten => 200,1,Answer
exten => 200,n,Playback(demo-echotest)
exten => 200,n,Echo()
exten => 200,n,Hangup
```

Echotest je nadefinován pro číslo 200. Při vytočení tohoto čísla se volajícímu zpráva přehraje. Volající může zanechat zprávu, k tomu slouží aplikace Echo. Poslední příkaz je položení hovoru.

Spuštění démona Asterisku je pomocí příkazu:

- `/etc/init.d/asterisk start`

Tímto se spustí démon Asterisku. Nyní máme Asterisk spuštěný, abychom ho mohli za běhu spravovat, musíme se dostat do konzole asterisku. To uděláme pomocí příkazu:

- `asterisk -rvvv`

Pomocí této konzole můžeme Asterisk přímo spravovat. Pokud stisknem Enter, můžeme hned zadávat příkazy. Pomocí příkazu *Help* si můžeme vypsat všechny známé příkazy a pak už je spravovat dle libosti. Pokud zadáme *Help <příkaz>,* tak nám konzole vypíše všechny informace a zadání tohoto příkazu[6][8].

Dříve, než vyzkoušíme volání na Asterisk, musíme nastavit hlasové menu. Pokud nemáme stáhnuté určité zvukové knihovny, nic se nám nepřehraje. Nyní nahrajeme své úvodní menu. Pro nahrání mého zvukového souboru jsem použil program Audacity. Použil jsem 8KHz, 16bits a výslednou nahrávku jsem uložil do formátu Windows PCM wav (16bit). Tento nahraný vzkaz musí mít určité parametry, proto ho ještě upravíme pomocí příkazu sox a jelikož mi wav soubor nefungoval správně, převedl jsem ho do formátu SLN. Příkaz:

- ***sox main-menu.wav –t raw –r 8000 –s -2 –c 1 main.sln***

Má nahrávka obsahuje: „Dobrý den, dovolali jste se na iDům, po stisknutí jedničky Vás přepojíme na Simplex pravý. Po stisknutí dvojky na Simplex levý. Po stisknutí trojky na tablo a po stisknutí čtyřky na telefon. Pokud si přejete zprávu zopakovat, stiskněte pětku.“

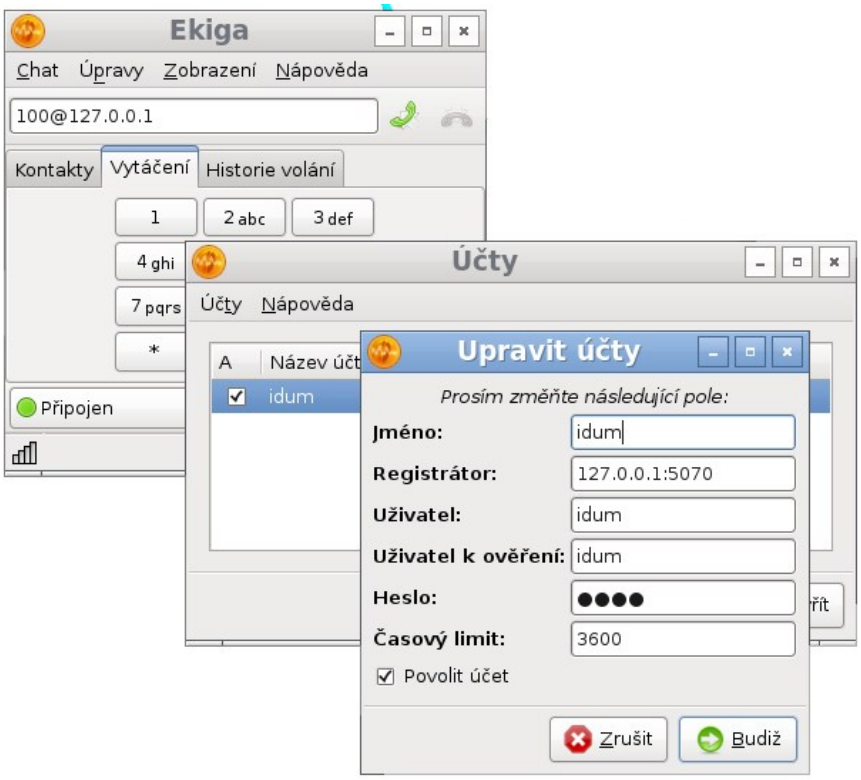
Stejným způsobem jsem nahrál další zvukové nahrávky, které použiji. A to „na slyšenou, děkujeme za zavolání“, „Volaný účastník není dostupný“, „Nyní Vás přepojíme na levý Simplex“, „Nyní Vás přepojíme na pravý Simplex“, „Nyní Vás přepojíme na Tablo“, „Nyní Vás přepojíme na Telefon“, „Volaný účastník hovor nepřijímá“, „Neplatná volba“.

Stejně jsme také provedli změnu formátu, pomocí programu SOX. Poté soubory zkopírujeme do složky pro audio soubory Asterisku. Složka se nachází /var/lib/asterisk/sounds. V konfiguračním souboru extensions.conf se jen ujistíme, že máme všechny zvukové nahrávky stejně pojmenované, jako jsou v dialplanu. Nezapomeňme, že po každé úpravě nastavení konfiguračních souborů je nutné restartovat Asterisk. To provedeme příkazem:

- ***/etc/init.d/asterisk restart***

Nyní můžeme otestovat naší IVR pomocí softwarového telefonu, který si nyní nainstalujeme.

Posledním nastavením bude nastavení softwarového telefonu, díky kterému si můžeme otestovat funkčnost nastavení našeho Asterisku, funkčnost spojení a případně otestování námi nastaveného dialplanu. Jak už bylo zmíněno, použiji softwarový telefon Ekiga, který podporuje protokol SIP.



Obrázek 4.1: Ekiga, nastavení účtu

Jak vidíme na obrázku 4.1, nastavení klienta je velice jednoduché. Jedná se o napsání stejných údajů, které jsme nastavili v Asterisku. Jako jméno jsme zadali idum. Registrátorem je adresa zvaná loopback, což znamená, že spojením na tuto adresu se spojíme s naším vlastním PC, také se dá nazvat adresou localhost. Adresa je tedy 127.0.0.1 a jelikož port, na kterém pracujeme byl zvolen 5070, napíšeme ještě za dvojtečku tento port. Zvolili jsme port 5070, protože testování spojení jsme zkoušeli na jediném notebooku, tzn., že by na portu 5060 (který bývá klasicky používán) probíhalo ke dvěma spojením. První spojení by bylo používané Asteriskem a druhé by využíval program Ekiga z kterého chceme volat. Uživatel a Uživatel k ověření je také „idum“. Časový limit jsme nechali tak, jak byl nastaven. Zaškrtli jsme kolonku povolit účet.

Při testování jsem však zjistil, že volání na Asterisk z Ekigy, kde oba programy jsou na stejném PC, není uskutečnitelné. Proto jsem vytvořil spojení dvou PC, kde jeden byl řídící PC iDomu společně s Asteriskem a druhý byl PC s Ekigou, kde jsem nastavil parametry, které jsem uvedl výše. Port 5070 jsem změnil zpět na defaultní port pro

TCP/UDP 5060. Z toho důvodu se do kolonky registrátor nemusela zadávat dvojtečka s portem, nýbrž pouze IP adresa PC, kde je Asterisk nainstalován.

Nyní máme nastaveného klienta a můžeme testovat náš dialplan a celkovou funkčnost námi nastavené ústředny. Po vytočení 100@192.168.1.1 (IP adresa PC, kde je nainstalovaný Asterisk) se dostaneme k hovoru na Asterisku a mělo by proběhnout spojení podle našeho dialplanu. Přesvědčit se o tom můžeme v konzoli Asterisku, kde nám objeví každé upozornění o tom, co se zrovna děje.

Ke konzoli se dostaneme, jak už jsem zmínil, pomocí příkazu Asterisk –rvvv, kde počet „v“ určuje úroveň Verbosity, což se dá nazvat „úrovní logování“. Vypisuje chybové hlášky, čím vyšší bude hodnota verbosity, tím podrobnější bude výpis programu. Verbosity taky způsobí to, že spuštěný Asterisk nepoběží na pozadí systému. Díky této konzoli snadno zjistíme chybu, či špatně zadaný příkaz. Konzole je důležitým nástrojem hlavně pro programátory, protože se, díky ní, najde snadněji chyba.

5 Konfigurace programu Asterisk pro spolupráci se zvukovou kartou řídícího PC

Konfigurace programu Asterisk, aby vysílala a přijímala zvukové informace skrze zvukovou kartu, se nastavuje v konfiguračním souboru oss.conf a nebo alsa.conf, což jsou sobory pro rozhraní zvukové karty. Nalezneme je ve složce /etc/asterisk. Tyto dva konfigurační soubory jsou velice podobné. Já jsem použil alsa.conf, kde jsem nastavil autoanswer na yes, což znamená, že pokud někdo zavolá, hovor je automaticky přijat. Dále context a extensions. Čímž je myšleno, jaké bude defaultní nastavení a kde má být hovor odkázán. U nás to je context idům a extensions 100, což je číslo iDomu. Poslední nastavením je nastavením vstupního a výstupního zařízení. V konfiguračním souboru to nalezneme pod input_device a output_device. Za tyto parametry jsem dosadil default pro input a default pro output, tedy defaultní nastavení vstupu a výstupu zvukové karty.

V případě volání prostřednictvím zvukovou kartu je volána v Asterisku extensions Dial(console/alsa) [13].

6 Skripty pro spojení a vyzvánění

Než přistoupíme k samotnému vytvoření vlastních skript, musíme nejprve pochopit skripty, která již byla vytvořena. Při připojení PC k iDomu jsme nejprve museli zadat pár příkazů v terminálu.

- ***sudo su*** - tento příkaz nás převede z práv uživatele do práv administrátora, u linuxu konkrétně se administrátor nazývá SuperUser
- ***apt-get install ssh*** – nainstaluje ssh
- ***apt-get install mc*** – nainstaluje midnight comannder, který se hodí pro práci se soubory
- ***cd /*** - změna složky; lomítko nám říká, že se dostaneme do rootu
- ***mkdir /temp*** – vytvoření složky „temp“
- ***chmod 777 /temp*** – změna práv pro složku „temp“
- ***cd iDum*** - změna složky, tentokrát přejdeme do složky „iDum“
- ***ls –all*** - nám vypíše všechny adresáře a soubory, včetně nastavených práv

Jelikož práva souboru iDům byla nastavena na nepovolené čtení souboru, museli jsme změnit práva:

- ***chmod 744 iDum*** – změna práv u souboru „iDům“

Změnili jsme práva u iDům, takže nyní můžeme tento soubor číst (spustit), ale předtím než si iDům spustíme, musíme si vytvořit odkaz, aby iDům věděl na co je připojen. My používáme připojení přes USB (kabel s redukcí USB na R232), proto nastavíme ttyUSB0. První však musíme odstranit existující symbolický odkaz na ttyS.

- ***rm ttyS –*** tímto příkazem odstraníme již běžící spojení
- ***ln –s /dev/ttyUSB0 ttyS*** – vytvoření symbolického odkazu

Nyní iDům můžeme spustit. Pokud se nacházíme ve složce /iDum/ zadáme tento příkaz:

- ***./iDum --USB*** – příkaz ke spuštění iDomu, který je podporován pro USB

Po spuštění iDomu se nám spustilo grafické rozhraní iDomu, kde můžeme nastavovat jednotlivá zařízení a spotřebiče, terminál s nastavením teploty a terminál iDům, kde se nám zobrazují jednotlivá upozornění, co iDům dělá. Terminál iDům je konzolí iDomu.

Terminál s řízením teploty a grafické rozhraní nepotřebujeme, takže ho můžeme ukončit. Nejdříve si nastíníme obsah skript. Skripty jsou v iDomu soubory, které nám říkají, co má jaké zařízení dělat. Skript si lze spustit pomocí příkazu:

- ***../SpustSkript NazevSkriptu.scr 0***

Musíme se však nacházet ve složce, kde skripty jsou. Ke skriptům se dostaneme ve složce /iDum/Scripts. V příkazu za NazevSkriptu.scr dosadíme název skriptu, který chceme spustit. Číslo na konci příkazu znamenají posílaná data.

Pro otevření kódu skriptu a jeho úpravu můžeme použít libovolný textový editor, já používám textový editor s názvem Gedit.

- ***gedit SimplexL.scr*** – spustí soubor SimplexL.scr v textovém editoru Gedit

Podívejme se nyní na skript pro levý Simplex a trochu si ho přibližme. Jak si můžete všimnout, skript je složen z příkazů a parametrů v závorce. Čísla ve tvaru RAM[0xXX] jsou pozice v paměti RAM. Každé číslo má svůj význam a seznam všech těchto čísel i s vysvětlením k čemu slouží, naleznete v /iDum/Scripts/Ram/Ram.txt. Do tohoto seznamu jsem přidal pozici v paměti 0x300, kterou jsem v souboru Ram.txt nazval VoIP. Tato paměť mi poslouží k identifikaci stavu o VoIP.

Ve skriptech při odesílání zprávy potřebujete znát ID zařízení, abyste věděli, kterému zařízení se zpráva odesílá. Všechny ID zařízení jsou definovány v souboru devid.tab, který naleznete v /iDum/Scripts/.

6.1 Skripty pro vyzvánění a spojení pomocí prvku „Simplex“

Simplex je zařízení umožňující obousměrnou komunikaci s jakýmkoliv komunikačním zařízením. V jeden okamžik smí hovořit pouze jeden člověk (princip vysílaček). Přepínání, kdy jeden může hovořit a druhý poslouchá je ruční, stejně jako u vysílaček. Na demonstračním panelu iDomu máme dva Simplexy nazývané SimplexL a SimplexP (levý a pravý). Tento Simplex má 2 tlačítka. Zelené tlačítko slouží pro komunikaci s druhým Simplexem. Před začátkem mé bakalářské práce nemělo červené tlačítko žádnou funkci (hovor s druhým Simplexem se přerušil po uvolnění zeleného tlačítka, jak už bylo zmíněno).

Obrázek 6.1: Simplex

SimplexP.scr a SimplexL.scr jsou názvy souborů skript pro Simplexy. Jelikož jsou nastaveny pro komunikaci pouze mezi sebou, bude nutné do těchto skript zasáhnout.

Než přistoupíme ke konkrétnímu tvoření skript, řekněme si základní body spojení mezi Asteriskem a iDomem. Začněme příkladem volajícího účastníka na SimplexL.

Dalo by se to zaznačit do několika bodů.

1. Asterisk otevře externí skript
2. SimplexL začne vyzvánět
3. stiskem zeleného tlačítka je otevřená audio linka na SimplexL
4. zvuková karta řídícího PC je zpřístupněna do audio linky na Simplexu
5. hovor je ukončen stiskem druhého tlačítka nebo volajícím účastníkem

Samozřejmě, počítejme s tím, že při stisku tlačítka může uživatel odpovídat volajícímu skrze Simplex, avšak při uvolnění zeleného tlačítka je možnost pouze hovor poslouchat, jak už bylo zmíněno.

Již v bodovém postupu hovoru dochází k problémům. Prvním je, že skripty SimplexL a SimplexP jsou momentálně určeny pouze pro komunikaci mezi sebou, což znamená, že pokud by začalo vyzvánění a stisklo by se tlačítko, Simplex by otevřel audio linku pro druhý Simplex. Stejný problém nastává v případě stisku tlačítka během hovoru. Proto je nutno do skriptu vložit podmínku. Při stisku tlačítka se musí zjistit, zda probíhá hovor nebo ne. V případě, že hovor neprobíhá, může se zpřístupnit audio linka pro druhý Simplex.

Dalším problémem je ukončení hovoru, jak hovor ukončit? Jelikož jsou na Simplexu dvě tlačítka a druhé tlačítko (červené) nemá, prozatím, žádnou funkci, můžeme toho využít.

Nyní se podívejme na konkrétní řešení skriptu pro Simplex. Pro vyzvánění jsem vytvořil nový skript z názvem VyzvaneniSimplexL.scr a VyzvaneniSimplexP.scr. Tento skript je volán jako externí skript pro Asterisk, v případě volby volání na jeden ze Simplexů. Skript má následující úkol. První zkontroluje, zda je audio linka volná, pokud ano, pokračuje dál, pokud ne, ukončí skript a Asterisku odešle zprávu, že volaný účastník hovor nepřijímá. V případě pokračování, kdy je audio linka volná, otevře ji pro svou potřebu, vloží do ní melodii, což je vyzváněcí melodie ve formátu mp3. Jakmile melodie dohraje, znamená, že hovor nebyl vyzvednut (nebylo stisknuto zelené tlačítko na Simplex), volá se skript KonecSimplexL.scr.

Skript KonecSimplexL.scr oznámí Asterisku, že volaný účastník hovor nepřijímá. Uzavře audio linku a skript se ukončí.

Nyní jsme si popsali, pokud hovor není přijat. Jak skript pokračuje, pokud je hovor přijat? Jestli je při vyzvánění stisknuto zelené tlačítko, je volán skript SimplexL.scr, který jsme museli upravit pro hovor tak, aby nedošlo k vytvoření komunikace mezi Simplexy a aby byl hovor přijat. Proto, při stisknutí zeleného tlačítka se skript ptá, zda probíhá hovor, vyzvánění, anebo nic. Skript zjistil, že probíhá vyzvánění, proto není otevřena komunikace mezi Simplexy, ale je přijat hovor, což znamená, že nechá otevřenou audio linku a Asterisku se oznámí přijetí hovoru. Při uvolnění tlačítka hovor stále probíhá, jen uživatel na Simplexu nemůže hovořit k volajícímu. Při opětovném stisku zeleného tlačítka to možné je. Jak už bylo zmíněno, Simplex funguje pouze na jednosměrné komunikaci, proto může hovořit jen jeden uživatel.

Při stisku červeného tlačítka se hovor ukončí tak, že Asterisku se oznámí zavěšení hovoru a skript zavře audio linku. Pro tento úkon se musel použit skript, který pracuje mezi iDomem a Asteriskem (více v kapitole 6.5 volání externího skriptu)

Skript rozeznává tři stavy a to: při vyzvánění je do paměti Ram 0x300 vložena hodnota 1, při hovoru hodnota 2 a pokud neprobíhá hovor ani vyzvánění, nabývá paměť hodnoty nula. Při stisku tlačítka se skript ptá na hodnotu paměti Ram 0x300 a podle toho odkáže pokračování skriptu do příslušné části. Ram 0x300 je mnou zvolená volná paměť, kterou jsem si určil pro VoIP. Aby Ram 0x300 nabývala při spuštění iDomu vždy hodnoty 0, museli jsme si to definovat ve skriptu Startup.scr, který také nalezneme ve složce /iDum/Scripts. V tomto skriptu jsme pomocí příkazu Oper vynulovali paměť 0x300. Oper je funkce pro matematickou operaci, jejíž syntaxe je: první argument, operace, druhý argument. Vykonaná operace mezi dvěma argumenty je zapsána do prvního argumentu. Prvním argumentem je RAM paměť 0x300. Druhým argument je 0. Operací je zde Load, což znamená, že přepíše paměť Ram 0x300 na 0. Konkrétně ve skriptu to vypadá následovně:

```
Oper(Ram[0x300],Load,0);
```

Pokud si otevřeme skript SimplexL.scr, vidíme zde několik příkazů. Vysvětleme si nyní, co jednotlivé příkazy znamenají. Skripty iDomu jsou celkem jednoduché a mají jednoduché syntaxe, o všech skriptech se můžete dočíst ve specifikaci iDomu.

Co se týče identifikačních značení u Simplexů, tak pro SimplexL je označení 6A a pro SimplexP je označení 6B. Toto označení se používá při odesílání zprávy. Zadává se v části zprávy „KOMU“. Pokud do zprávy zadáme do části „komu“ 6A, tak se zpráva

odešle na SimplexL. Seznam všech identifikačních označení najdete v devid.tab, který je ve složce /iDum/Scripts/, což jsem již zmiňoval v předchozí kapitole.

Stručně ke skriptům. Poznámky se vkládají do složených závorek, tedy {poznámka/komentář}. Celý skript je rozdělen do jednotlivých částí. Každá část má svůj název a za názvem dvojtečku, za dvojtečkou jsou už jednotlivé příkazy.

Základní funkcí je podmínka „if“. Pokud je podmínka splněna, přejde do definované části skriptu. Příkladem:

```
if(Msg[0],=,0x03,VyslanyByte);
Exit;
VyslanyByte:
…
```

Tato podmínka nám říká, pokud je typem zprávy msgByte pokračuje v kódu na VyslanyByte, pokud ne, skript se ukončí (Exit).

6.2 Skripty pro vyzvánění a spojení pomocí prvku „Telefon“

Na demonstračním panelu iDomu jsou dva telefony, stejný počet jako Simplexů. Tyto telefony mají klasické sluchátko s mikrofonem. Každý má pět neoznačených tlačítek, pouze jedno tlačítko, šesté, má označení, a to obrázek klíče. Jeden z telefonů je navíc obohacen o malý displej s kterým souvisí i malá kamera a ta je také umístěna na demo-panelu iDomu. Tlačítka měla před tvorbou mé bakalářské práce pouze dvě funkce. Při stisku prvního tlačítka se uskutečnil hovor na druhý telefon. U telefonu s displejem se po stisku druhého tlačítka otevřela video linka s kamerou a na displeji se zobrazilo to, co zrovna kamera snímala.

Skripty k telefonům v programu iDům nebyly. Vše probíhalo hardwarově, proto jsem si vytvořil nový skript s názvem Telefon1.scr. V tomto skriptu jsem nadefinoval prozatím pouze 3 stavy, kdy je telefon vyzvednut, zavěšen a kdy telefon dozvonil. V těchto třech částech dále budu implementovat spojení s Asteriskem a i následovné přerušení, respektive zavěšení hovoru.

Identifikační označení těchto dvou telefonů je u telefonu bez displeje CB, leží na levé straně uprostřed demonstrační desky. Telefon s displejem má označení CC a leží na pravé straně uprostřed demonstrační desky.

Co se týče skript pro telefon, tak se nejedná o nic složitého. Jedná se o zjištění ByteID a následovné odkázání, co má skript dál učinit. ByteID je typ zpráv, které jsou odesílány. 01 u ByteID znamená cmdCall, což je volání. ByteID 02 je cmdConnect, což je spojení. Další ByteID jsem zaznačil do tabulky (Tabulka 1.1). Samozřejmě existuje

více příkazů. V tabulce jsem zaznačil pouhý výběr nejpoužívanějších neboli námi použitých. Všechny příkazy naleznete v souboru Interkom.h.

Asterisk v IVR spouští skript VolaniNaTelefon.scr. Po spuštění tohoto skriptu začíná vyzvánět telefon s označením CB. Tento skript vyšle zprávu s příkazem cmdCall. Pokud uživatel telefon zvedne, otevře se skript Telefon1.scr, změní se zpráva v cmdConnect, čímž je skript odkázán do části TelefonConnect, kde nám vypíše na konzoli, že telefon byl vyzvednut. V této části skriptu dochází ke spojení s Asteriskem. Skript Asterisku oznámí, že Telefon byl přijat. Zpřístupní se spojení mezi telefon a Asteriskem. V případě zavěšení telefonu je volána část skriptu s názvem TelefonZavesil, která oznámí Asterisku, že uživatel telefon zavěsil. V opačném případě, kdy volající hovor zavěsí je oznámeno iDomu, že uživatel ukončil hovor a skript se odkáže do části TelefonZavesil. Zpráva je cmdHang a skript se uzavře.

Tabulka 1.1: *Seznam ByteID*

ByteID	Příkaz	Význam
00	cmdHang	Zavěšení
01	cmdCall	Volat (typ zařízení)
02	cmdConnect	Spojení, přestává vyzvánět
03	cmdOpen	Otevřit dveře (skupina)
04	cmdObsazeno	Obsazeno
05	cmdPresmeruj	Přesměrování (čísla)
06	cmdRingEnd	Konec vyzvánění
07	cmdCancelRing	Zrušení vyzvánění
09	cmdCode	Kód (code[6])
0A	cmdProdlouzeni	Prodloužení hovoru
0B	cmdCodeOK	Potvrzení kódu / informace o správném kódu
13	cmdNextCall	Současné volání
19	cmdSound	Přehrání zvuku

6.3 Skripty pro spojení pomocí prvku „Tablo“

Prvkem Tablo je myšleno zvonkové tablo, které je také obsaženo na demonstrační desce iDomu. Tohle konkrétní tablo je rozděleno do dvou částí. Na pravé

straně je trojmístný displej a klávesnice (číslice 0 až 9 s hvězdičkou a křížkem). Na levé straně zvonkového tabla je osm tlačítek určených k předvolbě (v praxi využívaných jako zvonky na jednotlivé domy či byty).

Princip fungování skriptu pro Tablo je postaven na stejném principu jako Telefony. Využívá posílání zpráv. Každá zpráva má svůj vlastní význam (viz Tabulka 6.1). S tím rozdílem, že Tablo nemá vyzvánění, jak je tomu u Telefonu. V případě, že bychom vyslali zprávu vyzvánění (cmdCall), iDům nás automaticky s Tablem spojí. Po vypršení časového limitu přehraje ukončující melodie a hovor zavěsí. Nenechme se zmýlit slovy „nemá vyzvánění“. Je to myšleno, že nemá vyzvánění, pokud voláme z Asterisku na Tablo, protože nemáme tlačítko k přijetí hovoru. Vyzvánění však nastane, pokud voláme z Tabla na předdefinované číslo (více v kapitole 6.5 Volání externích skript pro iDům).

V případě stisknutí jedničky na klávesnici zvonkového tabla nám Tablo začne volat na iDům (na řídící PC).

6.4 Skripty pro spojení volání z prvku „Tablo“ na předdefinované číslo

Skript pouze rozšiřuje již vytvořený skript Tablo. Skript je nastaven tak, že v případě, že uživatel chce volat na předdefinované číslo, stiskne číslici 4. Tablo zavolá externí skript na Asterisk, který vytočí předdefinované číslo, které jsem si určil 603. Pokud je v síti uživatel 603 přihlášen na svém softwarovém SIP telefonu, po stisknutí 4 mu telefon začne vyzvánět a nám Tablo také začne vyzvánět, což jsme provedli pomocí poslání zprávy s příkazem cmdSound (přehrání melodie/zvuku) s hodnotou 02, což je oznamovací tón (sndOznamovaci). Po přijetí hovoru volajícím na SIP telefonu je hovor přijat. Pokud volající z Tabla stiskne hvězdičku, hovor je ukončen a to tak, že Asterisku se oznámí zavěšení hovoru. Pokud hovor ukončí účastník na straně SIP telefonu, oznámí se iDomu konec hovoru na Tablu (zavolání skriptu KonecHovorNaiDum.scr, který už byl vytvořen před mou bakalářskou prací) a Asterisk samozřejmě pozná zavěšení hovoru, proto se mu nemusí ukončení hovoru oznamovat.

6.5 Volání externích skript pro iDům

V předchozích podkapitolách jsem uvedl, že bývá volán externí skript, který Asterisku oznámí zavěšení či spojení hovoru. Pojďme si nyní přiblížit, o jaký skript se jedná.

Systém iDům má funkci ExecFileParam(x,y), která volá externí program (program mimo iDům) s jedním parametrem a poté probíhající skript ukončí. Asterisk

má na své konzoli možnost, že je možné volat určité příkazy, kterými lze obsloužit probíhající hovor. Této skutečnosti jsem využil pro spojení mezi prvky a hovorem.

Princip fungování funkce ExecFileParam funguje právě na principu volání funkcí Asterisku přímo v konzoli Asterisku. Problém však nastává, pokud jsou v parametrech mezery. V tomto případě funkce není přijata a konzole iDomu vypíše upozornění na chybu. Tomuto problému jsem předešel pomoci volání shell souboru, kde volám shell soubor bez parametru a mám v něm definováné přímé obsloužení hovoru. Takto vyřešené volání externího programu již bez problému funguje.

Nyní si přibližme princip obsluhy hovoru, tedy obsah shell souborů.

6.5.1 HangUp

V případě, že uživatel iDomu chce hovor ukončit, musí se tato skutečnost oznámit i volajícímu, tedy zavolat na Asterisk funkci HangUp. Zadávání funkce HangUp v dialplanu a konzoli se poměrně liší. V dialplanu je funkce umístěna přímo u čísla, které chceme zavěsit, proto se ví, který hovor se má ukončit. Pokud bychom zadali funkci hangup přímo do konzole, Asterisk by nevěděl, který hovor má ukončit. Proto se musí za funkci HangUp zadat i číslo kanálu. Celá volaná funkce má tedy syntaxi:

- ***channel request hangup SIP/idum-00000001***

Na příkladu zavěšení vidíme ukončení hovoru s kanálem SIP/idum-00000001. Každý hovor má generované nové číslo kanálu (po restartu Asterisku se počítadlo generovaných kanálů vynuluje), je tedy nutné vždy zadat správné číslo kanálu, jinak by se příslušný hovor neukončil.

K vypsání běžících hovorů slouží příkaz:

- core show channels verbose

Pomocí tohoto příkazu si vypíšeme běžící kanály. Pomocí funkce GREP a AWK si uložíme jednotlivé kanály do proměnných, které jsou volány za hangup. Funkce „for“ obstará, aby byly postupně vypisovány jednotlivé kanály a jednotlivě je ukončovaly. Příkazem Echo vytvoříme výstup našeho hangup s číslem běžícího kanálu.

Tento soubor je volán: při hovoru na Simplexu, když uživatel stiskne červené tlačítko, a také je volán při zavěšení sluchátka telefonu.

6.5.2 Dial

Dial se volá na konzoli:

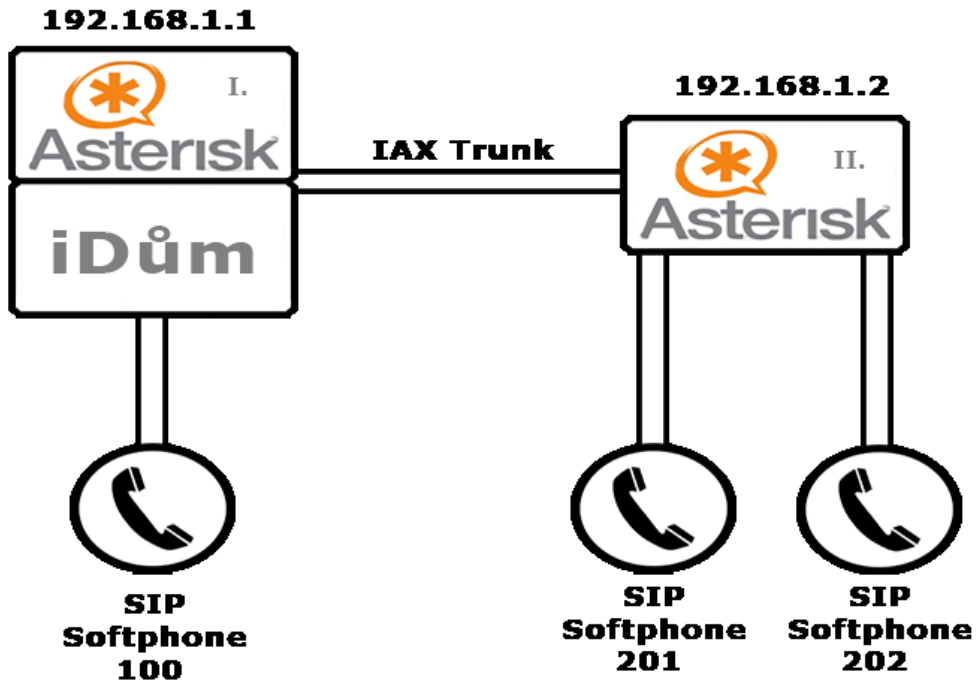
- ***console dial 603***

Jak vidíme na příkladu, příkaz dial je velice jednoduchý. Příkaz „console dial“ a číslo, které chceme volat. Zadáváni funkce Dial se liší od zadání v Dialplanu pouze tím, že v konzoli se před dial zadává console.

Tento soubor je volán při stisku čísla čtyři na Tablo a volá na předdefinované číslo, kterým je u nás číslo 603. Toto číslo je uživatel se SIP telefonem (softwarovým telefonem), který je registrovaný na Asterisku.

7 Konfigurace systému pro funkci ústředny pro spojení s ostatními prvky v síti používající VoIP

Konfigurací systému pro funkci ústředny pro spojení s ostatními prvky v síti využívající VoIP se myslí, že pokud jsou na stejné síti jiní uživatelé využívající VoIP, mohou se také spojit s iDomem. Tuto problematiku jsem demonstroval vytvořením dalšího Asterisku na jiném PC a spojením těchto dvou Asterisků. Jak můžete vidět na následujícím obrázku (Obrázek 7.1), první PC je PC s programem iDům a programem Asterisk, což je řídící PC iDomu. Druhý PC je PC pouze s Asteriskem. Tyto dva Asterisky jsou propojeny trunkem za použití IAX protokolu. Po zapojení této topologie bychom chtěli docílit spojením telefonů 201,202 a 100. Kde, bude schopen každý telefon volat na kterýkoliv jiný telefon zapojený v síti a to včetně na iDům.



Obrázek 7.1: Logická topologie zapojení dvou Asterisků a SIP telefonů

IAX trunk spojuje dvě ústředny stejného typu, u nás Asterisk, který je nainstalován na řídícím PC iDomu a Asterisk nainstalován na dalším PC. Smysl spojení těchto dvou ústředen je následovný. Pokud uživatel bude volat na číslo, které není v databázi jeho ústředny, ale ústředna ví, že se nachází na druhé ústředně, může tohoto využít a volat příslušnou extensions z druhé ústředny.

Pro přesnější představu si tuto problematiku můžeme vysvětlit na obrázku 7.1.. Pokud telefon s číslem 202 má zájem volat na číslo 100, vyšle příkaz na Asterisk. Asterisk z Dialplanu zjistí, že číslo 100 nemá v databázi, ale má ho definované jako odkaz na druhý Asterisk. Proto přes trunk zavolá druhý Asterisk a ten hovor obslouží.

Co se konkrétního nastavení týká, musíme nastavit obě ústředny. Nastavíme nově nainstalovanou ústřednu na druhém PC (II. Asterisk). S instalací Asterisku jsme se již seznámili v dřívější kapitole. Nastavení je obdobné jako u nastavování ústředny iDomu. V nastavení upravíme konfigurační soubor sip.conf.

```
[general]
context=ustredna
port=5060
udpbindaddr=0.0.0.0
svrlookup=yes
disallow=all
allow=alaw
allow=gsm
allow=ulaw
```

Jediné co se liší od nastavené ústředny iDům je název context, který jsem pojmenoval „ustredna“, protože nám demonstruje jakéhosi poskytovatele VoIP služby, přesněji řečeno nám pouze pro představu demonstruje poskytovatele VoIP. Dále nastavíme dva softwarové telefony. Telefony budou využívat protokol SIP, proto konfiguračním souborem, který budeme nastavovat, bude sip.conf. Zde si vytvoříme dva účty, jeden s číslem 201 a druhý s číslem 202. Přesné nastavení klientů jsme si uvedli ve čtvrté kapitole, proto konkrétní popsání nastavení klientů přeskočím.

Je také nutné vytvořit dialplan pro novou ústřednu. Ještě než se k tomu dostaneme, ujistěme se, že static je nastaven na yes a writeprotect na no. Nyní přikročíme k nastavení dialplanu. Náš dialplan bude mít dvě části. První část bude obsahovat jednotlivá čísla, která budou volána pomocí makra, a druhá část budou makra, kde vytvoříme základní konstrukci pro hovor. Nastavení extensions.conf:

```
[ustredna]
exten => 201,1,Macro(hovor,SIP/201)
exten => 202,1,Macro(hovor,SIP/202)
exten => 100,1,Macro(hovor,IAX2/idum/${EXTEN})
[IAXpeer]
exten => 201,1,Macro(hovor,SIP/201)
exten => 202,1,Macro(hovor,SIP/202)
[macro-hovor]
```

```
exten => s,1,Dial(${ARG1})
exten => s,2,Hangup
```

Jak vidíme na konkrétní ukázce dialplanu, hovor je specifikován v makru (kdy hovor vytočí a následně zavěsí) s voláním argumentu, který je přiřazen k příslušnému číslu. U čísla 100 je volán pomocí protokolu IAX iDům, respektive ústředna, která je pojmenována iDům, pak je volán jeho patřičný EXTEN. Context IAXpeer je contextem pro příchozí hovory z druhé ústředny.

Nyní máme nastavenou novou ústřednou, kterou si nazveme Asterisk II. Následuje zpřístupnění jedné ústředny s druhou. K tomu použijeme protokol IAX. Můžeme samozřejmě použít i protokol SIP pro trunk, ale aby nás nemátla přílišná konfigurace v konfiguračním souboru sip.conf, použijeme iax.conf, kde prozatím nemáme nastaveno nic. Konfigurační soubor iax.conf nalezneme tam, kde i ostatní konfigurační soubory jako je sip či extensinons. V iax.conf vytvoříme jakoby účet uživatele. Nicméně, nastavení se od klasického uživatele bude lišit. Nenechte se nastavením zmýlit, opravdu nastavujeme druhou ústřednu s názvem Asterisk II.

```
[idum]
type=friend
username=ustredna
secret=peer
auth=plaintext
host=192.168.1.1
context=IAXpeer
peercontext=IAXpeer
qualify=yes
trunk=yes
```

Tímto nastavením jsme vytvořili odkaz, respektive trunk na druhou ústřednu. Nyní musíme nastavit i ústřednu iDům a její příslušný dialplan. Nastavení je obdobné jako nastavení ústředny Asterisk II s tím rozdílem, že odkazy jsou na opačnou stranu.

V konfiguračním souboru iax.conf na ústředně Asterisk I (ústředna na řídícím PC iDomu) je třeba vytvořit trunk. Nastavení má podobnou syntaxi jako nastavení ústředny Asterisk II.

```
[ustredna]
type=friend
username=idum
secret=peer
auth=plaintext
host=192.168.1.2
```

```
context=IAXpeer
peercontext=IAXpeer
qualify=yes
trunk=yes
```

Trunk je nastaven, nyní se přesuneme k již vytvořenému dialplanu, který mírně poupravíme. Dialplan zůstane stejný jako doposud, pouze je tam vloženo makro hovor s odkazy na případné spojení hovorů. Ukázku kódu zde nebudu vypisovat, protože je syntaxí naprosto stejný, akorát odkaz na druhou ústřednu není u čísla 100, nýbrž u čísel 201 a 202.

Takto vytvořený trunk by měl fungovat, tak jak jsme si popsali na začátku. Na podobném principu fungují ústředny VoIP poskytovatelů, kteří mají svou ústřednu, kde přidávají uživatele, kteří si mohou mezi sebou volat.

Závěr

Technologie VoIP je v současné době velice lukrativní alternativou klasické telefonní přípojky. Jelikož VoIP lze využívat i na PC či chytrém telefonu, můžeme volat za velice přijatelné ceny, a to i do zahraničí.

Cílem mé bakalářské práce bylo ukázat možnost implementace pobočkové ústřednu do systému inteligentního domu, která je schopna telefonovat pomocí technologie VoIP a také schopna komunikovat s prvky iDomu.

Systémem, kde jsem měl tuto technologii implementovat byl iDům vyvíjený na katedře Telekomunikační techniky VŠB-TU Ostrava, na kterém jsem vytvořil pobočkovou ústřednu využívající technologii VoIP. Pobočková ústředna využívá hlasové menu pro přesměrování hovoru na jednotlivé prvky iDomu a také poskytuje volání z prvku Tablo na předdefinované číslo a to vše za spolupráce Asterisku a zvukové karty řídicího PC. Skripty se mi podařily navrhnout a nastavit do fáze, že všechny body mého zadání jsou plně funkční. V současném stavu demonstračního panelu iDomu nejde demonstrovat výstup z iDomu na Asterisk. V budoucnu lze demonstrovat při hardwarovém zásahu do výstupu mikrofonu rozhraní řídicího PC, protože konektor není uzpůsoben pro výstup zvukové karty notebooku, který jsem použil.

Mou bakalářskou prací jsem také dokázal demonstrovat poměrně jednoduchá nastavení skriptů pro iDům, což nám dokazuje dobrou dostupnost nastavení pro uživatele, kteří neměli s nastavením těchto skript dřívější zkušenosti. A také jsem demonstroval možnost obohatit iDům o pobočkovou ústřednu.

Jako řídící PC jsem použil notebook se systémem Linux Debian a programem iDům s verzí z roku 2011. Celý systém i program iDům probíhal z USB flash disku. Jelikož jsem tuto bakalářskou práci řešil z notebooku připojeném na rozhraní iDomu a nikoliv z řídícího PC připojeného přímo na demonstrační desce iDomu, nemůžu potvrdit, ale ani vyvrátit funkčnost vytvořené ústředny přímo na řídícím PC na demonstrační desce. Z teoretických poznatků by problém neměl nastat, ale praxe může být jiná. Co se týče demonstrace sedmé kapitoly, tedy spojení pro ostatní prvky v síti využívající VoIP. Pro demonstraci této kapitoly jsem použil dvě PC, které se nenacházejí u demonstrační desky iDomu, proto tuto problematiky nelze u iDomu odzkoušet

Použitá literatura

[1] Ing. JAHELKA, Michal, Ph.D. *Dokumentace systému iDům*. Ostrava 2013. Specifikace iDum.pdf. Vědecká práce. VŠB-TU Ostrava.

[2] Bc. ŠIMEK Jaroslav. *Ovládání inteligentního domu mobilním telefonem*. Ostrava 2012. Diplomová práce. VŠB-TU Ostrava.

[3] SOVA Lukáš. *VOIP – Hlasová komunikace v IP sítích, VOIP GSM*. České Budějovice 2006. Absolventská práce. Soukromá vyšší odborná škola a Obchodní akademie s.r.o.

[4] Jak funguje VOIP? *Earchiv.cz* [online]. [citace 9. 4. 2014] Dostupné z: http://www.earchiv.cz/b06/b0401001.php3

[5] WIJA Tomáš, ZUKAL David, VOZŃÁK Miroslav. *Asterisk a jeho použití*. Praha 2005. Technická zpráva. CESNET.

[6] MEGGELEN Jim Van, MADSEN Leif, SMITH Jared, *Asterisk: The Future of Telephony, 2nd Edition*. Sebastopol 2007. O´Reilly. ISBN – 10: 0-596-51048-9

[7] Pobočková VoIP ústředna Asterisk. Root.cz [online]. [citace 9. 4. 2014] Dostupné z: http://www.root.cz/serialy/pobockova-voip-ustredna-asterisk/

[8] Asterisk: VoIP ústředna. *Abclinuxu.cz* [online]. [citace 9. 4. 2014] Dostupné z: http://www.abclinuxu.cz/serialy/asterisk-voip-ustredna

[9] JACKSON Benjamin, CLARK III Champ. *Asterisk Hacking*. Burlington 2007 Syngress.

[10] Asterisk Sound File. *Voip-info.org* [online]. [citace 15. 4. 2014] Dostupné z: http://www.voip-info.org/wiki/view/Asterisk+sound+files

[11] VOZŃÁK Miroslav, ŘEZÁČ Filip. *Asterisk teorie a praxe*. Ostrava 2011. Vysoká škola báňská – Technická univerzita.

[12] Asterisk hardware recommendations. *Voip-info.org* [online]. [citace 5. 5. 2014] Dostupné z: http://www.voip-info.org/wiki/view/Asterisk+hardware+recommendations

[13] Asterisk config alsa.conf. *Voip-info.org* [online]. [citace 5. 5. 2014] Dostupné z: http://www. voip-info.org/wiki/view/Asterisk+config+alsa.conf

Seznam příloh

Součástí bakalářské práce je CD.

Adresářová struktura přiloženého CD:

/iDum/ - obsahuje shell soubory pro komunikaci Asterisku a iDomu

/iDum/Scripts/ - obsahuje skripty pro iDům, které byly pro účel mé bakalářské práce pozměněny nebo vytvořeny

/asterisk1/ - obsahuje konfigurační soubory programu Asterisk vytvořené pro pobočkovou ústřednu Asterisk na řídícím PC

/asterisk2/ - obsahuje konfigurační soubory programu Asterisk vytvořené pro pobočkovou ústřednu Asterisk na druhém PC